APPENDIX D

Josephus Problem

In real-world applications, especially in the operating system, multiple activities have to be performed. The biggest issue is not just performing these activities but also completing them in minimum time. The Johnson's algorithm is used in such applications where an optimal order of execution of different activities has to be determined.

Consider a problem that consists of independent tasks $T_1, T_2, \ldots, T_n$ and two independent processes P_1 and P_2. If it is specified that P_1 must be completed before P_2, then Johnson's problem can be given as:

Step 1 Determine P_1 and P_2 times for each task.

Step 2 Make two queues, Q_1 and Q_2 where Q_1 is formed at the beginning of the schedule and Q_2 is formed at its end.

Step 3 For each task, analyse P_1 and P_2 times to determine the smallest time. If P_1 is the smallest time, then insert the corresponding task at the end of Q_1. Otherwise, insert the corresponding task at the beginning of Q_2. In case of a tie, take P_1 as the smallest time.

Note If there is a tie between multiple P1 or multiple P2 times, select the first task in the list.

Consider the table given below, which specifies the tasks and time it takes to complete processes P_1 and P_2.

TASK	TIME TO PERFORM P_1	TIME TO PERFORM P_2
0	17	6
1	24	12
2	5	8
3	14	10
4	11	8
5	14	11

Now, for each task, analyse P_1 and P_2 times to determine the smallest time. Tasks with P_1 time less than P_2 are assigned to the head of Q_1, other tasks are assigned to the tail of Q_2.

TASK	TIME TO PERFORM P_1	TIME TO PERFORM P_2	MINTIME	LOCATION
0	17	6	6	TAIL
1	24	12	12	TAIL

2	5	8	5	HEAD
3	14	10	10	TAIL
4	11	8	8	TAIL
5	14	11	11	TAIL

Sort the table using the MINTIME field.

TASK	TIME TO PERFORM P_1	TIME TO PERFORM P_2	MINTIME	LOCATION
2	5	8	5	HEAD
0	17	6	6	TAIL
4	11	8	8	TAIL
3	14	10	10	TAIL
5	14	11	11	TAIL
1	24	12	12	TAIL

There is an alternative implementation strategy which states that if LOCATION has the value TAIL then the task is added at the front of Q_2. Once all the tasks are assigned, HEAD and TAIL can be concatenated to create the final complete QUEUE.

When calculating the efficiency of the Johnson's algorithm, we see that the data is processed as one observation at a time, thereby taking `O(n)` time where `n` is the volume of the data. Then the data must be sorted which will again take at least `O(n × log n)` time. Since, `O(n × log n)` term dominates. Johnson's algorithm gives an optimal schedule in `O(n log n)` time.